Week 8 Database Assignment

Solving food wastage through sustainable consumption and production of the SDG 2030

Part 1

Problem:
Food wastage is not just an ethical and economic issue; it is a critical challenge to achieving sustainable development. In line with SDG 12 of the 2030 Agenda, this project, 'Solving Food Wastage Through Sustainable Consumption and Production,' aims to address inefficiencies in the food system, reduce waste at every stage—from farm to fork—and foster responsible practices that safeguard resources for future generations. By implementing data-driven strategies and sustainable innovations, we can reshape how food is produced, consumed, and valued, ensuring that no plate is left empty while preserving our planet's resources.

--The problem to be addressed using data is:

- **Tracking and reducing food waste in urban households.**

Data can be used to track food consumption, expiration, and waste patterns, and analyse trends to provide insights into where the highest levels of waste occur. This information could then be used to design interventions, such as smart alerts for food expiration or tips for reducing waste. The data can also be used to minimise the amount spent on certain food products that are easily spoiled and those that are deemed essential by the users to avoid unnecessary waste.

Part 2: Database design

The database design section using SDG 12 (Responsible Consumption and Production), focusing on food waste tracking:

A. Entity-Relationship Diagram (ERD)

The entities based on the problem are:

1. User: Tracks information about individual users or households.
2. Food Item: Represents food items purchased by the user.
3. Waste records : Records the instances where food is wasted, along with the reason.
4. Consumption Logs: Tracks food consumption by the user.
5. Expenses: tracks the amount spent on food either consumed or wasted.

Here is a description of the relationships:

- A User can have multiple Food Items.
- A food category can have multiple Food Items.
- A Food Item can have multiple entries in the Waste record (if wasted) and the Consumption Log (if consumed).
- A User can also have multiple Waste records and Consumption Logs for different food items.

ERD Breakdown:

1. User (1) ⟶ (M) Food Item
2. Food Item (1) ⟶ (M) Food categories
3. Food Item (1) ⟶ (M) Waste records
4. Food Item (1) ⟶ (M) Consumption Log

B. **Schema:** Write SQL statements to create the database schema based on your ERD.

There are several statements that are used to create the database based on the above-mentioned ERD.

1. To create the database one can use this command:

-- Create the database

-- CREATE DATABASE IF NOT EXISTS FoodWasteManagement;

2.The use the database using this command:

-- Use the database

-- USE FoodWasteManagement;

3.Created a tables with columns within database that align with the SDG 12, using the following commands.

-- Table: Users (Represents individuals or entities tracking food consumption and waste)

-- CREATE TABLE Users (

-- user_id INT AUTO_INCREMENT PRIMARY KEY,

-- username VARCHAR(50) NOT NULL,

-- email VARCHAR(100) NOT NULL UNIQUE,

-- created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP

--);

-- Table: FoodItems (Represents the different types of food items being tracked)

-- CREATE TABLE FoodItems (

-- food_id INT AUTO_INCREMENT PRIMARY KEY,

-- food_name VARCHAR(100) NOT NULL,

-- category VARCHAR(50), -- e.g., "Vegetables", "Fruits", "Dairy", etc.

-- created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP

--);

-- **Table for storing expenses**

-- CREATE TABLE Expenses (

-- expense_id INT AUTO_INCREMENT PRIMARY KEY,

-- user_id INT NOT NULL, -- Foreign key to reference the Users table

-- food_id INT NOT NULL, -- Foreign key to reference the FoodItems table

-- expense_type ENUM('Waste', 'Consumption'), -- Type of expense: either waste or consumption

-- amount DECIMAL(10, 2) NOT NULL, -- Expense amount

-- expense_date DATE NOT NULL, -- Date the expense was recorded

-- FOREIGN KEY (user_id) REFERENCES Users(user_id),

-- FOREIGN KEY (food_id) REFERENCES FoodItems(food_id)

--);

-- Table for Consumption (Tracks the quantity of food consumed by each user)

-- CREATE TABLE Consumption (

-- consumption_id INT AUTO_INCREMENT PRIMARY KEY,

-- user_id INT, -- Foreign key to Users table

-- food_id INT, -- Foreign key to FoodItems table

-- consumption_date DATE NOT NULL,

-- expense DECIMAL(10, 2),

-- quantity_consumed DECIMAL(10, 2) NOT NULL, -- Quantity in kilograms

-- FOREIGN KEY (user_id) REFERENCES Users(user_id) ON DELETE CASCADE,

-- FOREIGN KEY (food_id) REFERENCES FoodItems(food_id) ON DELETE CASCADE,

-- created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP

--);

-- Table: WasteRecords (Tracks the quantity of food wasted by each user)

-- CREATE TABLE WasteRecords (

-- waste_id INT AUTO_INCREMENT PRIMARY KEY,

-- user_id INT, -- Foreign key to Users table

-- food_id INT, -- Foreign key to FoodItems table

-- waste_date DATE NOT NULL,

-- expense DECIMAL(10, 2),

-- quantity_wasted DECIMAL(10, 2) NOT NULL, -- Quantity in kilograms

-- reason VARCHAR(255), -- Reason for food waste (e.g., "Expired", "Spoiled")

-- FOREIGN KEY (user_id) REFERENCES Users(user_id) ON DELETE CASCADE,

-- FOREIGN KEY (food_id) REFERENCES FoodItems(food_id) ON DELETE CASCADE,

-- created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP

--);

-- -- Table: FoodCategories (Tracks different categories of food for classification)

-- CREATE TABLE FoodCategories (

-- category_id INT AUTO_INCREMENT PRIMARY KEY,

-- category_name VARCHAR(50) NOT NULL

--);

C. **Sample Data:** Populate the database with relevant sample data.

The date can be inserted in the graphs using the following commands:

1. Insert Sample Users

-- INSERT INTO Users (username, email) VALUES

('Alice', 'alice@example.com'),

('Bob', 'bob@example.com');

2. Insert sample food items

INSERT INTO FoodItems (food_name, category, created_at) VALUES

('Apples', 'Fruits', NOW()), -- food_id = 1

('Milk', 'Dairy', NOW()), -- food_id = 2

('Bread', 'Bakery', NOW()), -- food_id = 3

('Yoghurt', 'Dairy', NOW()); -- food_id = 4

3. Insert categories into FoodCategories

INSERT INTO FoodCategories (category_name) VALUES

('Fruits'),

('Dairy');

4. Insert sample consumption records

INSERT INTO Consumption (user_id, food_id, consumption_date, quantity_consumed) VALUES

(1, 1, '2024-09-10', 1.5), -- User 1 consumed 1.5 kg of Apples

(1, 2, '2024-09-11', 0.5), -- User 1 consumed 0.5 litres of Milk

(2, 4, '2024-09-12', 1.0), -- User 2 consumed 1 kg of Yoghurt

(2, 1, '2024-09-13', 1.0); -- User 2 consumed 1 kg of Apples

5. Insert sample waste records

INSERT INTO WasteLog (user_id, food_id, quantity_wasted, waste_reason, waste_date) VALUES

-- User 1

(1, 1, 3.0, 'Spoiled', '2024-09-08'), -- User 1 wasted 3 kg of Apples

(1, 2, 1.0, 'Expired', '2024-09-13'); -- User 1 wasted 1 litre of Milk

6 Insert sample of expenses

INSERT INTO Expenses (user_id, food_id, expense_type, amount, expense_date) VALUES

(1, 101, 'Waste', 50.75, '2024-09-01'), -- User 1, Food 101, waste-related expense

(2, 102, 'Consumption', 25.50, '2024-09-02'), -- User 2, Food 102, consumption-related expense

(3, 103, 'Waste', 70.00, '2024-09-03'), -- User 3, Food 103, waste-related expense.

Part 3: SQL Programming

There are several commands that were applied to retrieve and analysis of datasets within the database.

A, Data Retrieval: Write SQL queries to retrieve relevant data based on your problem definition.

For data retrieval the following commands were used:

Total Quantity of Food Consumed by Each User

```
-- SELECT
--     c.user_id,
--     u.username,
--     SUM(c.quantity_consumed) AS total_quantity_consumed
-- FROM
--     Consumption c
-- JOIN
--     Users u ON c.user_id = u.user_id
-- GROUP BY
--     c.user_id, u.username;
```

Food Items Consumed by a Specific User

```
-- SELECT
--     f.food_name,
--     c.quantity_consumed,
```

-- c.consumption_date

-- FROM

-- Consumption c

-- JOIN

-- FoodItems f ON c.food_id = f.food_id

-- WHERE

-- c.user_id = 1;

B. Data Analysis: Write SQL queries to analyse data and generate insights related to your SDG problem.

Total Quantity of Each Food Item Consumed

-- SELECT

-- f.food_name,

-- SUM(c.quantity_consumed) AS total_quantity_consumed

-- FROM

-- Consumption c

-- JOIN

-- FoodItems f ON c.food_id = f.food_id

-- GROUP BY

-- f.food_name;

Total Waste by Food Item

-- food wasted by item

-- SELECT

-- f.food_name,

-- SUM(w.quantity_wasted) AS total_quantity_wasted

```
-- FROM
--     wasterecords w
-- JOIN
--     FoodItems f ON w.food_id = f.food_id
-- GROUP BY
--     f.food_name;
```

Food Consumption Ratio

```
SELECT
    f.food_name,
    COALESCE(SUM(c.quantity_consumed), 0) AS total_quantity_consumed,
    COALESCE(SUM(w.quantity_wasted), 0) AS total_quantity_wasted
FROM
    FoodItems f
LEFT JOIN
    Consumption c ON f.food_id = c.food_id
LEFT JOIN
    wasterecords w ON f.food_id = w.food_id
GROUP BY
    f.food_name;
```

Part 5: Integration and Testing

Integration: Step by step guide

Step 1: Prepare Your Data

1. **Source Data**: Ensure that your data is in a suitable format for importing (e.g., CSV, TXT, or Excel).
2. **Clean Data**: Check for and resolve any inconsistencies in the source data, such as missing values, duplicate entries, or formatting issues.

Step 2: Open Excel

1. **Launch Excel**: Open Microsoft Excel on your computer.

Step 3: Import Data

1. **Go to the Data Tab**: Click on the "Data" tab in the Excel ribbon.
2. **Select Data Source**:
    - For CSV or text files, click on "Get Data" → "From File" → "From Text/CSV".
    - For Excel files, click on "Get Data" → "From File" → "From Workbook".
3. **Browse and Select File**: Navigate to the location of your data file and select it.
4. **Import Wizard**:
    - For CSV/TXT: An import wizard will appear. You can preview the data and set delimiters (if applicable).
    - For Excel: Select the desired worksheet and range to import.
5. **Load Data**: Click "Load" to import the data into Excel. You can choose to load it into a new worksheet or an existing one.

Step 4: Ensure Data Consistency

1. **Check Data Types**: Verify that the data types (e.g., dates, numbers, text) are correctly formatted.
    - Select the columns and change the format (Home → Number) if necessary.
2. **Use Data Validation**:
    - Go to the "Data" tab and click on "Data Validation".
    - Set rules for data entry (e.g., drop-down lists, number ranges) to ensure consistency in future data entries.
3. **Remove Duplicates**:

- Select the range of data.
- Go to the "Data" tab and click on "Remove Duplicates".
- Choose the columns to check for duplicates and click "OK".

4. **Fill in Missing Values**:
    - Identify and fill any missing values appropriately (e.g., using averages, zeros, or forward filling).
5. **Use Conditional Formatting**:
    - Highlight inconsistencies or anomalies.
    - Select the range, go to "Home" → "Conditional Formatting" to set rules for highlighting specific conditions.

Step 5: Save and Document Your Work

1. **Save the Excel File**: Save your work regularly to avoid data loss.
2. **Document the Process**:
    - Create a separate documentation sheet within the Excel file or a separate Word document.
    - Describe each step taken during the import process.
    - Include any rules set for data validation and consistency checks performed.

Testing: Done on excel spreadsheet and an interactive dashboard is found there.

Part 6: Presentation

Pitch deck done using canva and the slides can be found on PDF labelled **Database analysis.**